REMARKS/ARGUMENTS

Regarding Amendments

In the specification, the "Cross Reference to Related Applications" section has been amended to update the status of referenced applications. These corrections are of a clerical nature and do not add "new matter".

Claims 1-9, 13-15, 17-39, 43-45, 47-65, 69-71, and 73-82 are now pending.

No claims stand allowed.

Claims 10-12, 16, 40-42, 46, 66-68, and 72 have been cancelled, without prejudice or disclaimer.

The Amendment also contains minor changes of a clerical nature. No "new matter" has been added by the Amendment.

The First 35 U.S.C. §103 Rejection

Claims 1-9, 13-15, 25-27, 29-39, 43-45, 55-65, 69-71, and 81-82 stand rejected under 35

U.S.C. §103(a) as being allegedly unpatentable over Yellin et al.[1] in view of Levy et al.[2], among which claims 1, 25-27, 29, 30, 31, 55-57, 81, and 82 are independent claims.[3]

This rejection is respectfully traversed.

According to the M.P.E.P.,

> To establish a *prima facie* case of obviousness, three basic criteria must be met. First there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, not in the applicant's disclosure.[4]

Furthermore, the mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination.[5]

Claim 26

Claim 26 recites:

> A computer implemented process for managing exceptions throwable during execution of two or more methods in one or more classes by a virtual machine on a resource-constrained device, each method included in a class and including protected code and an exception handler array defining exception handlers associated with the method, the protected code of the two or more methods stored in a first portion of a package according to an ordering, the individual exception handler arrays combined in block form and forming a single exception handler table for the two or more methods, the exception handler arrays positioned in a second portion of the package based on the ordering, the process comprising:
> searching the exception handler table when an exception is thrown while executing one of the methods including locating a first matching exception in the single exception

[1] USP 5,761,513.
[2] USP 6,092,147.
[3] Office Action dated September 9, 2003, ¶ 3.
[4] M.P.E.P. §2143.
[5] *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990).

handler table.

The Examiner states:

> As to claim 26, Yellin discloses managing exceptions throwable during execution of methods in classes (line 66 column 1 to line 28 column 2) by a virtual machine (lines 11-57 column 3), each method (each method, line 13 column 3) including protected code (protected code block, line 63 column 3) and an exception handler array defining exception handlers associated with the method (the code for the exception handlers, line 14-15 column 3), the protected code of methods stored in a first portion (thrownable, error and exception, 122) of a package (Java class file, line 20 column 3) according to an ordering, combining in block form (Fig. 1) the exception handler arrays for methods into a single exception handler table (one table of exception handlers for all the methods in a class, lines 16-18 column 3); the exception handler arrays positioned in a second portion of the package (ThreadDeath to NoSuchMethodError, Fig. 2), searchng the exception handler table (found in a tree search, lines 42-44 column 3) when an exception is thrown (an exception is thrown, line 41 column 3) while executing one of the methods (while executing the protected code block, line 65 column 3) including locating a first matching exception in the single exception handler table (the enclosing exception handlers that is applicable to the thrown exception, lines 42-44 column 3). However, Yellin does not explicitly disclose a resource-constrained device.
>
> Levy teaches implementing a virtual machine on a resource-constrained device (Fig. 1). It would have been obvious to apply the teachings of Levy to the system of Yellin because this provides a reduction in the overall memory size and an increase in the overall processing speed of the virtual machine as disclosed by Levy (lines 33-55 column 2).[6]

The Applicants respectfully disagree. Claim 26 recites in part storing protected code of two or more methods in a first portion of a package according to an ordering, and positioning exception handler arrays associated with the methods in a second portion of the package *based on the ordering*.

The Examiner further states:

> Applicant argued that Yellin does not show correlation between the loading order of protected code and exception handler arrays (Remarks, first paragraph page 26). In response, Fig. 1 of Yellin reference clearly show the links of one method to other

[6] Office Action ¶ 3.

> methods through the use of protected code wherein the exception handlers of these methods are all arranged in one table (lines 16-18 column 3) ...[7]

The Applicants respectfully submit the Examiner's response does not address the claimed limitations. In support of the Examiner's statement, the Examiner refers to FIG. 1 of Yellin et al., which is referenced in the background section of Yellin et al. In the same sentence, the Examiner also refers to lines 16-18 col. 3 of Yellin et al., which are in the detailed description section of Yellin et al. and which make no reference to FIG. 1 of Yellin et al.

Also, contrary to the Examiner's statement, FIG. 1 of Yellin et al. does not show exception handlers of methods arranged in one table. Rather, FIG. 1 of Yellin et al. shows EndOfFile Exception Handler 100 that is associated with method C.1 (104) located at the end of the code for method C.1 (104). Yellin et al. also shows EndOfFile Exception Handler 114 that is associated with method E.X1 (112) located at the end of the code for method E.X1 (112). The exception handlers for Method C.1 (104) and Method E.X1 (112) of Yellin et al. are not shown in the same table. Nor is the code for Method C.1 (104) and the code for Method E.X1 of Yellin et al. shown to be stored within a first portion of a package according to an ordering. Nor are the exception handlers (100, 114) of Yellin et al. shown to be positioned within a second portion of the package *based on the ordering*, as recited in claim 26.

The Examiner refers to the following portion of Yellin et al.:

> For the purposes of this invention it makes no difference whether there is a separate table of exception handlers for each method, or just one for all the methods in a class.[8]

[7] Office Action ¶ 5.

Note the above says nothing about storing code for the methods in a first portion of a package according to an ordering. Nor does it disclose positioning exception handlers associated with the methods in a second portion of the package *based on the ordering* as recited in claim 26.

The Examiner is reminded that the mere absence from a reference of an explicit requirement of a claim cannot be reasonably construed as an affirmative statement that the requirement is in the reference.[9] For this reason, the 35 U.S.C. § 103 is unsupported by the art and should be withdrawn.

Thus, Yellin et al., whether considered alone or combined with or modified by Levy et al., does not teach storing protected code of two or more methods in a first portion of a package according to an ordering, and positioning exception handler arrays associated with the methods in a second portion of the package *based on the ordering*. Accordingly, the 35 U.S.C. §103 Rejection rejection of claim 26 based on Yellin et al. in view of Levy et al. is unsupported by the art and should be withdrawn.

Claim 1

Claim 1 recites:

A computer implemented process for managing exceptions throwable during execution of methods in one or more classes on a resource-constrained device, each method including protected code and an exception handler array defining exception handlers associated with the method, the process comprising:
defining an ordering for the methods, the ordering defining the placement of the protected code in a first portion of a package; and

[8] Yellin et al. col. 3 lines 16-18.
[9] *In re Evanega*, 829 F.2d 1110, 4 USPQ2d 1249 (Fed. Cir. 1987).

combining in block form the exception handler arrays for the methods into a single exception handler table, the combining comprising positioning the exception handler arrays in a second portion of the package based on the ordering.

Claim 1 includes limitations similar to claim 26 and thus must be allowable for at least the same reasons as claim 26.

Dependent Claims 2-9 and 13-15

Claims 2-8 and 13-15 depend from claim 1. Claim 1 being allowable, claims 2-8 and 13-15 must also be allowable.

Claims 27, 29-39, 43-45, 55-65, 69-71, 81, and 82

Claims 27, 29, and 30 are system claims corresponding to method claims 1, 25, and 26, respectively. Claims 1, 25, and 26 being allowable, claims 27, 29, and 30 must also be allowable.

Claims 31-39, 43-45, 55, and 56 are program storage device claims corresponding to method claims 1-9, 13-15, 25, and 26, respectively. Claims 1-9, 13-15, 25, and 26 being allowable, claims 31-39, 43-45, 55, and 56 must also be allowable.

Claims 57-65, 69-71, 81, and 82 are means-plus-function claims corresponding to method claims 1-9, 13-15, 25, and 26, respectively. Claims 1-9, 13-15, 25, and 26 being allowable, claims 57-65, 69-71, 81, and 82 must also be allowable.

Accordingly, it is respectfully requested that the 35 U.S.C. §103 Rejection rejection of claims based on Yellin et al. in view of Levy et al. be withdrawn. In view of the foregoing, it is respectfully asserted that the claims are now in condition for allowance.

The Second 35 U.S.C. §103 Rejection

Claims 17-24, 28, 47-54, and 73-80 stand rejected under 35 U.S.C. §103(a) as being allegedly unpatentable over Yellin et al. in view of Levy et al., and further in view of Bak et al.[10], among which claims 17, 28, 47, and 73 are independent claims.[11] This rejection is respectfully traversed.

Claim 17

Claim 17 recites:

> A method minimizing the amount of storage required for a runtime stack when executing a program, the runtime stack maintained at runtime during the execution of the program on a resource-constrained device for storing one or more frames where each frame includes a return pointer to an invoking method that called a currently executing method in the program, the method comprising:
> defining an ordering for methods included in the program, each method including protected code and an exception handler array defining exception handlers associated with the method, the ordering defining the placement of the protected code in a first portion of a package;
> combining in block form exception handler information for the methods included in the program into a combined exception handler table, the combining comprising positioning the exception handler arrays in a second portion of the package based on the ordering; and
> locating and searching the combined exception handler table when an exception is thrown during execution of one of the methods to locate the exception handler information without requiring the storage on the runtime stack of a pointer to the exception handler information.

[10] USP 6,009,517.
[11] Office Action dateed September 9, 2003, ¶ 4.

The Examiner states:

> As to claim 17, note the discussion of claims 26 above. However, Yellin as modified does not disclose a return pointer. Bak discloses a stack with frames wherein each frame includes a return pointer (line 52 column 2 to line 41 column 3). It would have been obvious to apply the teachings of Bak to the system of Yellin as modified because this allows exceptions propagate through the execution stack for handling by the appropriate exception handier, even when the functions were written in different languages and the format of the exceptions are different as disclosed by Bak (lines 52 column 2 to line 7 column 3).[12]

Claims 17, 28, 47, and 73 include limitations similar to claim 1. As noted above, Yellin et al. and Levy et al. do not make claim 1 obvious. For the same reasons, Yellin et al., Levy et al., and Bak et al. cannot be said to make claims 17, 28, 47, and 73 obvious.

Claims 28, 47-54, and 73-80

Claim 28 is a system claim corresponding to method claim 17. Claim 17 being allowable, claims 28 must also be allowable.

Claims 47-54 are program storage device claims corresponding to method claims 17-24. Claims 17-24 being allowable, claims 47-54 must also be allowable.

Claims 73-80 are means-plus-function claims corresponding to method claims 17-24. Claims 17-27 being allowable, claims 73-80 must also be allowable.

[12] Office Action dateed September 9, 2003, ¶ 4.

In view of the foregoing, it is respectfully asserted that the claims are now in condition for allowance. It is respectfully requested that the 35 U.S.C. § 103 rejection of claims based on Yellin et al. in view of Levy et al. and further in view of Bak et al. be withdrawn.

Request for Allowance

It is believed that this Amendment places the above-identified patent application into condition for allowance. Early favorable consideration of this Amendment is earnestly solicited.

Request for Entry of Amendment

Entry of this Amendment will place the Application either in condition for allowance, or at least, in better form for appeal by narrowing any issues. Accordingly, entry of this Amendment is appropriate and is respectfully requested.

If, in the opinion of the Examiner, an interview would expedite the prosecution of this application, the Examiner is invited to call the undersigned attorney at the number indicated below.

Respectfully submitted,
THELEN REID & PRIEST, LLP

Dated: January 9, 2004

John P. Schaub
Reg. No. 42,125

Thelen Reid & Priest LLP
P.O. Box 640640
San Jose, CA 95164-0640
Tel. (408) 292-5800
Fax. (408) 287-8040